DOCUMENT RESUME

ED 345 521 FL 020 216

AUTHOR Hart, Robert S.
TITLE Current Trends in Computer-Based Language Instruction.
PUB DATE 87
NOTE 14p.; For the journal as a whole, see FL 020 212.
PUB TYPE Information Analyses (070) -- Journal Articles (080)
JOURNAL CIT IDEAL; v2 spec iss p63-74 1987

EDRS PRICE MF01/PC01 Plus Postage.
DESCRIPTORS Applied Linguistics; *Authoring Aids (Programing); Classroom Techniques; *Computer Assisted Instruction; *Computers; *Courseware; Educational Trends; *Error Analysis (Language); *Feedback; Grammar; Instructional Development; Language Processing; Second Language Instruction; Second Language Learning; Teacher Developed Materials; Trend Analysis; Word Processing

ABSTRACT

A discussion of computer-based language instruction examines the quality of materials currently in use and looks at developments in the field. It is found that language courseware is generally weak in the areas of error analysis and feedback, communicative realism, and convenience of lesson authoring. A review of research under way to improve lesson quality suggests preferred solutions in each of these areas. In the case of error analysis, where feedback is important to effective and efficient learning, the best hope for systematic, grammar-sensitive error analysis seems to be based on parsing techniques borrowed from natural language processing. Communicative realism can be enhanced by both simulation designs and interactive video, even though materials using the approaches are still rather crudely designed. Finally, a promising approach to convenient lesson authoring involves adaptation of word processing techniques to create analogies with textbook authoring. A brief list of references is included. (Author/MSE)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED345521

Current Trends in Computer-Based Language Instruction

Robert S. Hart

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Wayne Dickerson

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

U.S. DEPARTMENT OF EDUCATION
Office of Educational Research and Improvement
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

X This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official OERI position or policy.

FL 020 219

BEST COPY AVAILABLE

IDEAL 2, 1987

CURRENT TRENDS IN COMPUTER-BASED LANGUAGE INSTRUCTION

Robert S. Hart

Current computer-based language lessons are weak in the areas of error analysis and feedback, communicative realism, and convenient lesson authoring. This paper reviews research underway to improve lesson quality in each area. The best hope for systematic grammar-sensitive error analysis seems to be based on parsing techniques borrowed from natural language processing. Both simulation designs and current work with interactive video offers greater communicative realism. One approach to convenient lesson authoring involves adaptation of word processing techniques to create analogies with textbook authoring.

It is not news that the attempt to make computer-based language instruction into a national educational reality faces a wide range of problems. Most schools now have computers, but sometimes only one or two per hundred students. When there are machines, foreign language students are often denied access in favor of mathematics and science classes. Textbook publishers are jumping into the language courseware market, but they want to keep costs down, and the bulk of what they do falls into the category of electronic workbook exercises. Freelance authors write for the lowest common denominator of hardware in order to increase their market. There is powerful new hardware, but it is not clear that most schools will be able to afford it any time soon. And, to complete this litany of difficulties, many language teachers simply do not believe that computer-based instruction really improves on the textbook and paper-and-pencil techniques it is designed to supplant.

This skeptical attitude of teachers, which arguably accounts in large part for the remaining problems, cannot be shrugged off simply as the results of ignorance, conservatism, or technophobia, since those same teachers have been quite willing to adopt computer technology (for example, word processing) when it is clearly superior to the pre-computer alternative. Most foreign language teachers, by inclination as well as necessity, are connoisseurs of new teaching methodologies, and if they are not convinced that computer-based language instruction is a superior tool it is probably because the evidence just is not there.

Current courseware does in fact lack features which most teachers consider crucial to efficient language learning, notably:

Error analysis and feedback: Teachers want programs that identify students' written grammar errors, and that mark them the way that paper-and-pencil exercises are marked.

Communicative realism: Teachers would like more realistic lessons which make interaction with the lesson seem like a real language use situation.

Convenient lesson authoring: Existing options for creating lessons are cumbersome. Teachers need an "electronic ditto machine," useful for quickly manufacturing exercises targeted directly at their own classes and syllabi.

FL 020 216

As long as these features are lacking, most teachers are simply not going to fight to put language courseware into their classrooms, no matter how hard the sales pitch. Fortunately, new microcomputer hardware and software now commonly available provide an improved set of tools for attacking these problems and a great deal of work intended to remedy these shortcomings is now underway. The remainder of this paper is devoted to reviewing some of this work.

ERROR ANALYSIS

Whenever the student has to type a response longer than a word or two, the possibility of multiple grammar errors exists. Feedback which locates and identifies the errors actually present should (presumably) lead to more efficient learning than the ever-popular "no, try again," or some generic hint at the correct answer. Instructors have been requesting this capability nearly as long as language courseware has existed, but attempts to provide it have all fallen short in various ways.

The PLATO system "error markup" (Figure 1) typifies one approach to feedback. This scheme describes errors in terms of editing operations such as insertion of extra words, omission of required words, movement of words to the right or left of their correct condition,

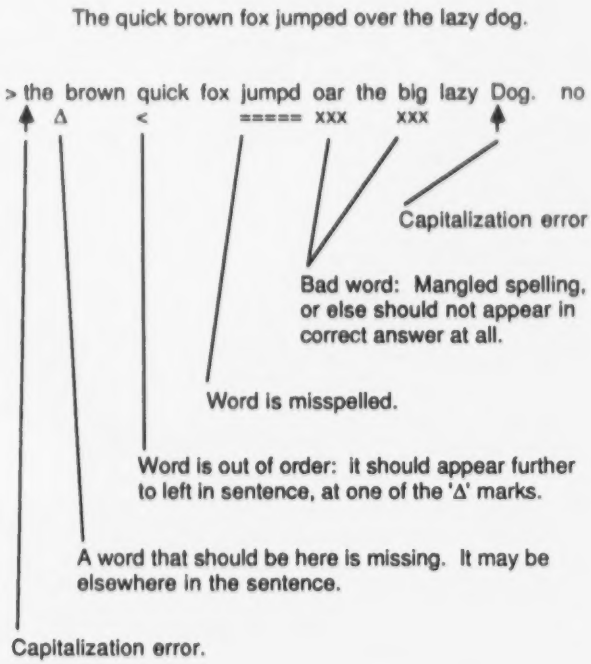
The quick brown fox jumped over the lazy dog.
> the brown quick fox jumpd oar the big lazy Dog. no
Δ
<
===== xxx xxx
Capitalization error
Bad word: Mangled spelling, or else should not appear in correct answer at all.
Word is misspelled.
Word is out of order: it should appear further to left in sentence, at one of the 'Δ' marks.
A word that should be here is missing. It may be elsewhere in the sentence.
Capitalization error.

Figure 1. PLATO error markup

incorrect upper- or lower-case letters. The kinds of errors present are shown by a standard system of graphic markup symbols. Other programming languages go even further in this direction. For example, EnBASIC[1], a dialect of BASIC designed explicitly for application to computer-based instruction, provides special markup symbols for accent errors and for insertions, omissions, or transpositions of individual letters. The popular DASHER[2] authoring system uses a scheme similar in spirit, but points of disagreement are indicated by dashes or carets.

In all these systems, the error markup is done automatically as a normal part of response evaluation, requiring no attention from the lesson author. The meaning of the graphic symbols can be learned quickly. A good student can often get sufficient information from the markup to diagnose and correct the grammar errors. In some instructional contexts, this is sufficient. For language teachers who teach explicit grammar rules, however, this style of error analysis has a severe drawback, since it makes no reference at all to the rules implicated in each error, and in fact marks errors just the same for Swahili or Russian as for English. The student is left to infer which grammatical rule was violated, and this is often expecting too much even of a good student.

One obvious way to accomplish this is to anticipate the incorrect structures which might occur in the student's response and search for each of them. In the following example, adapted from Marty's (1981) *Written French* series on PLATO IV, the student is asked to translate the English sentence

She translated the last ten pages for us.

Here the corrected answer is

Elle nous a traduit les dix dernières pages.

Suppose that the student has responded by typing

Elle a traduit le dernier dix pages pour nous.

Marty's program conducts the error analysis by searching through the student's response for each of the strings at the left, and by giving the error message on the right if it is found:

dernières dix	*Order of Adjective / cardinal number*
pour nous	*Prep + pn --> indirect object pn*
dernier	*Gender error: "page" is feminine*

Notice that the items searched for are literal strings. The error messages are telegraphic descriptions of grammar rules used in Marty's textbook. A student who finds them too cryptic can press the HELP key to receive a more detailed explanation.

From a language teacher's viewpoint, the anticipated error approach provides some nice features lacking in the PLATO error markup, since grammar errors can be identified according to whatever taxonomy the author prefers, and feedback messages can be given whatever phrasing he or she wishes. However, this way of proceeding also entails some serious disadvantages. For one thing, the author must hand-analyze each item individually in order to specify anticipated errors--a tedious and expensive undertaking if the volume of material is large. More fundamentally, different kinds of errors within a single item can

interact in such a way that specifying all possible combinations causes a combinatorial explosion. As a result, string search is practical only when the answer is very short or when the author specifies just a few of the most important errors.

The risk of combinatorial explosion is alleviated when pattern descriptors of greater generality are available, as is the case, for example, when a string processing language like SNOBOL is used to program the analysis. But this is at best an evasive maneuver. The real problem with string matching is that a program written this way does not have enough knowledge about the grammar or morphology of the target language, or about typical student errors, to produce an intelligent error description.

The ideal error analysis algorithm would mark an item, as much as possible, the way a teacher would mark it on paper, but do it automatically, without requiring a hand analysis. This implies that the algorithm should have the following three properties:

1. It should be *grammar specific*, identifying the particular grammar rules implicated in various types of errors;

2. The error messages actually shown to the student should be *adaptable to the teacher's preferred terminology* and presentation conventions.

3. It should be *completely automatic*, requiring no item-by-item programming.

The third property is not strictly necessary, but it is very desirable since it makes implementing new materials much faster and easier, hence less expensive.

Although Levine (1973) did some early exploratory work on this approach, no one seems to have followed his lead. Grammar sensitive error analysis has been successfully implemented in working instructional programs only for rather limited domains of of morphology, for example, Cully's (1979) Latin materials, originally done on the Delaware PLATO system but since adapted to the IBM-PC, and Davidson's Russian verb lesson[3] on the IBM-PC. Although several projects are under way, there is still no program which handles a large set of morphology and syntax.

The Language Learning Laboratory at the University of Illinois, Urbana-Champaign, is now building a system intended to approximate this ideal. A general solution clearly requires incorporating information about the target language grammar, which we accomplish by borrowing techniques from natural language parsing and adapting them for the case of ill-formed input.

We require (as do the authors of all the other schemes discussed above) that a model or correct answer be available beforehand. Analysis of free responses, where no correct answer is available, would require the program to construct an appropriate model before performing the error analysis proper. Determining "what the student was trying to say" generally requires the integrated use of grammatical, semantic, and pragmatic information and presents very difficult theoretical problems. In addition, to do such computations dynamically will likely require computing resources far exceeding those available to individual students in the foreseeable future. To obtain further simplification, we also require that the desired error descriptions be obtainable from grammatical analysis alone, possibly supplemented by some knowledge about what kinds of written errors language learners are likely to make; no semantic information is utilized.

The system, schematized in Figure 2, is now being implemented on an IBM-AT using GOLDEN COMMON LISP. The target language in this pilot version is French. The two inputs are the model, that is, the correct answer, and the student's response. The correct answer, which is assumed to be well formed, is first submitted to a morphological analysis, then parsed using standard parsing techniques. The output is a parsing tree with feature-labelled nodes, plus a table of role registers which label the grammatical functions of various constituents.

The student's response is also morphologically analyzed, then parsed, but since it may be ill-formed, the parsing must be highly error tolerant. Several means are employed to obtain such tolerance. First, the parser ignores most inflectional morphology and parses mainly in terms of roots; second, deviant rules which represent common syntactic misconstructions are included in the grammar. (When an incorrect construction is identified, it is of course marked as such.)

Morphological analysis utilizes a root dictionary and an inflection dictionary, both containing morphological and syntactic features. To assure that incorrect inflections can be identified and remediated, these dictionaries must contain more than the usual amount of information about inflectional morphology. Specifically, relational links are introduced "upward" to root (citation) forms, and "downward" from each root to its various stems and suppletive forms, including (when appropriate) ill-formed variants. This allows, for example, recognition that the verb *allent* in *Ils allent au cinéma* is an analogical (but incorrect) formation, and that it corresponds to the correct form *vont* in the correct answer *Ils vont au cinéma*. The latter information is necessary for successful matching at the next stage.

The two parsing trees are then submitted to a matcher which is responsible for matching up constituents from the two trees and for identifying and describing deviations from a perfect match. Since the trees may vary greatly in structure, matching proceeds bottom-up, starting with the heads of the various constituents to secure reliable correspondences. When a direct, word-for-word match cannot be made, the matcher can try various rearrangement rules, each of which corresponds to a particular type of error. For example this matching rule

((?X pn io) = (pp ("pour" prep) (?X pn));

("*pour* should not be used to indicate a pronoun io"))

states that an indirect object pronoun in the model may be matched with a suitable prepositional phrase in the student's response, thus identifying errors of the type

...nous a traduit ...	(model)
a traduit ... pour nous	(student response)

A suitable error message can be associated with each matching rule. For example, the prepositional construction is judged by some French speakers to be not truly incorrect. If the instructor chooses to take this position, the error message could be modified to read *"Nous a traduit* is also acceptable and might be preferable," or something of the sort. Intra-sentence errors, such as concordance errors, which do not require reference to the model, can be handled by examining patterns within the response parsing tree.

BEST COPY AVAILABLE

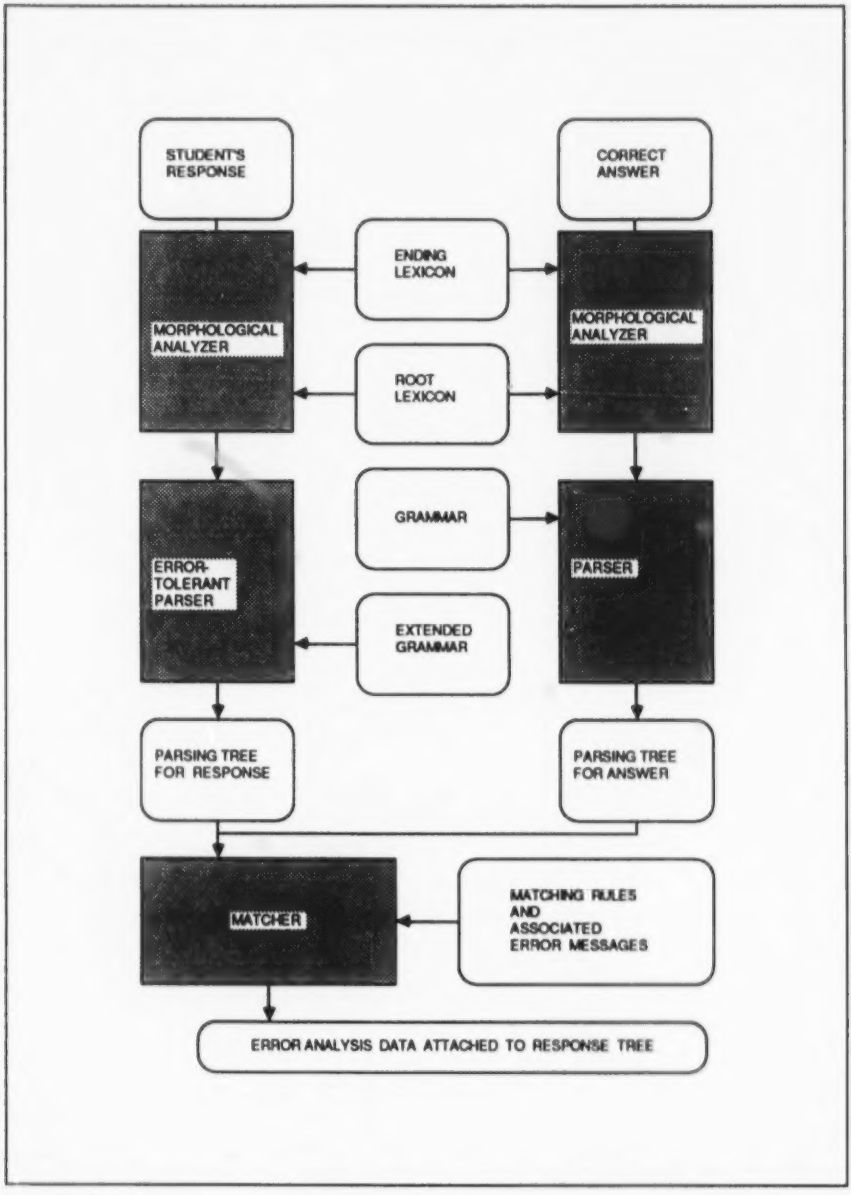

Figure 2. Block diagram of error analysis program.

The approach outlined here has decisive advantages over simple string-matching techniques. The fact that constituent structure is available to control matching makes mismatches less likely, and pattern descriptions of greater abstractness can be specified to the matching algorithm, allowing for a more compact and perspicuous statement of relevant grammar. A dictionary and grammar must of course be created, something not required for string matching, but they need be implemented only once for each language. Considerable flexibility in the phrasing of feedback can be obtained by simply changing the messages attached to the various matching rules. If this does not provide sufficient tailoring to individual pedagogical approaches, the error taxonomy can be redefined by changing the matching rules, without rewriting the dictionary or parser. A reasonable degree of modularity in the program design is thus maintained.

Several groups, for example those involved with the Athena project (Kramsch et al., 1986; Morgenstern, 1986), are working on the problem of providing grammatically sensitive error feedback. It is virtually certain that the next few years will see functioning error analysis systems.

A REALISTIC COMMUNICATION ENVIRONMENT

The phrase "realistic communication environment" simply means that computer lessons should use language in the same w, a correspondence which has several aspects:

Conversational use: The computer responds to and uses language the way a human being would, meaningfully, and with regard for the rules of conversational interaction.

Contextual use: Language is used within a rich auditory and visual environment.

Communicative use: Language is used in a meaningful and instrumental way, as part of some ongoing activity.

Encouraged by the communicative competence philosophy, attempts to produce communicatively realistic programs are currently a major theme in language courseware development. Some ambitious projects, such as MIT's project Athena, are trying to work toward all aspects of communicative realism in parallel. A few reservations are in order, however, about the feasibility of conversational language. First, getting a program to really converse within anything other than a trivial domain of discourse, if it can be done at all, will require very advanced artificial intelligence techniques, and hardware resources far in excess of anything likely to be generally affordable for standard instructional use in the foreseeable future. Nor are speech understanding programs likely to accommodate anything close to normal speech any time soon. Writing is not normally a conversational activity, and any program which depends on typing as the sole mode of inputing language is far from ideal as a conversational medium.

Obviously, this does not imply that we should give up trying to develop conversational programs. It does imply however that they will probably remain showcase items for a long time to come. For the moment, other aspects of communicative realism, for instance, meaningful and instrumental use, seem more tractable to computer implementation, and are is likely to yield greater payoff if pursued.

Simulation designs offer a promising approach to the meaningful and instrumental use of language. One well-known foreign-language example is *Cartels & Cutthroats*[4], a French-language economics simulation which has marketing and finance as its domain of discourse. The user is made president of a manufacturing corporation which must compete in the open market. The president is given an initial set of resources and must react to economic events outside his or her control, such as fluctuating interest and inflation rates, price changes by suppliers and competitors. etc. He or she in turn makes budgeting and pricing decisions that will either make or break the company. Several students can play at once, and converse (presumably in French).

Information is both tabular and verbal, the way it would actually reach an executive officer's desk. The user produces very little French, since input is generally numerical, but is exposed to a modest subset of commercial French. And the eighteen page instruction manual is also written in French. Failure to comprehend this French-language information prevents the user from functioning successfully in the simulation and soon results in bankruptcy.

Another example of this sort is *The Would-be Gentleman*[5], which simulates some of the sociology and economics of the France of Louis XIV. The user plays the role of a young bourgeois trying to climb, by judicious management of his resources, into the ranks of the rich and noble. Although written in English, the program could easily be converted to French, and would be an excellent vehicle for learning about French culture through the medium of the French language.

Although simulations can be very complex programs, they do not have to be. *Cartels & Cutthroats* is actually quite crude, and there is no reason that it could not be more highly developed, even within the very constrained APPLE II environment. Of course, the program was not designed for teaching French, and would be better suited to language teaching if it were provided with a glossary, paraphrase options, or other language teaching functions. This redesign would face a problem endemic to simulation programs, namely, that realistic simulation works against graded introduction of vocabulary and grammar constructs. As a result, keeping control of the language content without trivializing the simulation is difficult.

Context, both oral and visual, are obviously an important aspect of realism, and the authors of language courseware usually try to supply it by incorporating computer graphics into their lessons These, however, are generally not very satisfactory in terms of cost, space requirements, or final quality. High quality animation is even more difficult.

The new videodisc technology appears to offer a neat, relatively cheap solution to all these difficulties. There are language courseware authors who believe that micro/video interfacing will revolutionize language lessons by allowing presentation of culturally authentic material, nuances of body language, etc.

Given adequate time and money, it is certainly possible to assemble plenty of interesting "authentic" video material. How to use it in a way that exploits the interactive nature of the computer is another matter. It is useful to remember that a videodisc is a fixed data base, and that one can retrieve on request only from among a fixed set of visual items. The effectiveness of the lesson is a function not only of the video content, but also of the retrieval strategy employed. The strategies in use so far seem to be rather simple. The student makes choices to branch to different scenes, as in *Montevidisco* (Gale, 1983), or asks a question which the program answers by retrieving a video scene with the relevant information, as with some work done by project Athena.

These designs do not exhaust the possibilities of videodisc; perhaps they are not even the most useful ones. Presently the Language Learning Laboratory is investigating the use of video graphics as a context for efficiently teaching concrete vocabulary. We have done a little pilot work in this direction. In some formats, the computer discusses a visual scene and, while doing so, points to the objects being discussed. In others, the computer orally describes objects and asks the student to point them out. A game format called *Cherchez l'homme* is directed at teaching the vocabulary of clothing and personal appearance. The program displays pictures of several French men, then gives an oral description of one of them in terms of clothing and general physiognomy. The student's task is to point out the man being described. Since most of the details in the description could apply to more than one of the men pictured, a student must listen carefully and understand well to make a correct identification. When a wrong choice occurs, the program gives remedial feedback, in French, by explaining orally why the choice is wrong (e.g., "No, this man has blond hair, not black, and his hands are not in his pockets").

These formats are still rather crude. Our goal is to make them like some classroom exercises: very fast, very interactive, with constant exposure to audio and video, and with as much repetition as is necessary to internalize the vocabulary and learn the grammar. The next step, of course, is to introduce action scenes as a device for teaching verb vocabulary and verbal concepts such as tense and aspect.

We also plan to investigate designs which depend on categorizing and interrelating the library of scenes in various ways. For example, a brown book could be contrasted with a blue one, or one girl with several girls. Words could be taught by generalization from a number of specific examples, and temporal subordinators could be taught using sequences of action scenes. If the student indicates that he can not understand what is being said about a scene, we could use a common classroom technique and retrieve simpler paraphrase messages or video scenes which are simplified or differently focused. (This kind of paraphrasing is not practical, of course, unless we have a way to select the audio message independently of the picture. It is an important technical problem to provide this capability; either speech synthesis or CD-ROM may offer a solution.) If the program also maintains a learning history for the student and uses it to control the difficulty level of presentation, we might end up with instruction which looks somewhat like a computer-based version of the natural method.

The power of this sort of design comes from having a lot of knowledge about the scenes, and a program which can use that knowledge to select video information according to a strategy which will lead to efficient comprehension learning. Determining what kind of knowledge to include and how to represent it is an open problem which will require input from both instructional design theory and linguistics.

EASY-TO-USE AUTHORING FACILITIES

Some expert opinion holds that lesson authoring, like textbook writing, is too complex and time-consuming a task for the average teacher, and should be left to individuals who have special expertise in producing computer-based instruction. In this view, really good materials can result only from a long-term (and very expensive) development effort. This view ignores the fact that microcomputers are ideal devices for doing desktop publishing of both on-line and off-line materials. Teachers want authoring facilities for the same reason they want access to mimeo machines: because they frequently find it desirable to produce ephemeral materials which are hand-crafted to individual syllabi, students, or teaching styles,

BEST COPY AVAILABLE

and which can be redone or modified to reflect changes in current events, student interests, or classroom techniques. Such materials typically forego complex instructional design in favor of simple formats already familiar to both instructor and student. But authoring must be quick, because the materials intrinsically have a short lifetime, and because teachers typically have little free time to devote to lesson authoring.

General purpose programming languages cannot satisfy this need for speed and efficiency because their flexibility makes them complex and difficult to learn and use. Some language-specific authoring systems attempt to avoid these difficulties by adopting a template-based, fill-in-the-blank approach. The author selects one from a small set of exercise formats and types in some language items. The system then takes care of presenting the items according to the selected format. In this case, authoring simplicity is achieved at the cost of flexibility. The formats are more or less fixed, and modifying the program or adding new formats is likely to be difficult or impossible.

Certain courseware authoring systems attempt to maintain flexibility by compromising between general programming languages and template systems. Unfortunately, currently available systems of this type are typically oriented toward programmed learning designs and stress screen layout. Even when they provide for truly easy screen design (which is not always the case), control of sequencing remains primitive. Screens must generally be implemented in small pieces. Getting all the pieces together in a desired sequence may involve several menus and editors--at best an awkward process, at worst, impossible. Using such an authoring tool to create a simple vocabulary drill turns into a sizable project.

In part, these problems result from actual limitations of capabilities, but clumsy user interfacing plays an even greater role. The study of human-machine interaction has shown that an effective interface must provide a *user model* of how the interface works. This means, in essence, that the designer should find an activity that the user can already do well, and construct the interface so that it functions, as much as possible, in the same way. This idea may seem like common sense, but a quick survey shows that it is not much applied by the designers of authoring systems.

Teachers who are beginning to write computer-based lessons naturally enough use as a model the writing of non-computer lesson materials such as classroom exercises and textbooks. At this point, it will be tempting for the computer sophisticate to sigh and patiently explain once more the radical differences between programming a lesson and writing a textbook. But human factoring suggests that stressing this distinction may be bad strategy. Instead, we should perhaps do just the opposite, namely, facilitate the identification by making the authoring process resemble ordinary writing as closely as possible.

The Language Learning Laboratory is now exploring this possibility by building a language authoring system designed along these lines. As described by Cheng (1986), the user interface will be a word processor which is function, not procedure, oriented interactive, not reactive, and direct manipulation, not command, driven

The word processor will provide simple sequencing control in the form of an outliner facility. Questions, answers, and feedback will be interactively prompted without requiring the user to leave the word-processing environment. Various drill and exercise formats may be specified and their data typed directly into the text, just as would be done with textbook exercises. The intent is that all authoring be done by creating text in a (heavily prompted) word-processor environment. All the specifications for the lesson will appear as text, which is a familiar and non-threatening medium for new authors.

Of course the lesson itself will generally be highly interactive. The completed lesson text is submitted to an interpreter which generates appropriate screen displays, collects and judges responses, keeps scores, status, and learning history information, and controls branching and routing. This system is being implemented on the IBM-PC. Display is done in bit-map graphics to support a full variety of non-Roman writing systems, pictorial material, and a variety of type sizes and fonts.

I do not want to give the impression that a word-processor model solves all the problems of author/program interaction. As in any direct-manipulation approach, branching presents a troublesome issue. For many common instructional operations, variables and conditionals seem to be the most natural way of controlling program flow. Introducing such control mechanisms into the text reintroduces aspects of a procedural programming language and destroys the "what you see is what you get" property of direct manipulation. Devising a representation which will be natural both for screen display and for flow of control information remains a central problem for the proposed authoring language.

CONCLUSION

The research reviewed above suggests the outline of a new generation of language courseware. While not yet reaching the level of truly "intelligent" instruction, it will be substantially more sophisticated than the drill-and-practice materials we have now, and it will invalidate many of the criticisms leveled at current lessons. Micro hardware and software presently on the market or under development will provide delivery systems for these new lessons at something approximating an affordable price. The net result will be a much stronger case for classroom deployment of computer-based foreign language instruction.

THE AUTHOR

Robert S. Hart is Assistant Professor of Humanities and Associate Director of the Language Learning Laboratory at the University of Illinois at Urbana-Champaign. Address correspondence to G70 Foreign Languages Building, 707 S. Mathews St., Urbana, IL 61801.

NOTES

PLATO is a registered trademark of the Control Data Corporation. EnBASIC is a registered trademark of the Computer Teaching Corporation. GOLDEN COMMON LISP is a registered trademark of Golden Hill Computers, Inc. Courseware Authoring System is a registered trademark of McGraw-Hill, Inc. Cartels & Cutthroats is a registered trademark.

[1]EnBASIC, authored by A. Avner, S. Smith, and P. Tenczar, is distributed by Compress, PO Box 142, Wentworth, NH 03282.

[2]DASHER, authored by J. Pusack, is distributed by Conduit, PO Box C, Oakdale, IA 52319.

[3]The BMCAI program for error analysis of Russian verbs is presently under development by Dan Davidson of Bryn Mawr College. For information contact BMCAI, Bryn Mawr College, Department of Russian, Bryn Mawr, PA 19010.

[4]Cartels & Cutthroats is distributed by Le Professeur, PL Box 301, Swanton, VT 05488.

[5]The Would-be Gentleman, authored by Ruth Lougee for the Apple Macintosh, is distributed by Kinko's Academic Courseware Exchange, 4141 State Street, Santa Barbara, CA 93110.

REFERENCES

Cheng, C. C. (1986). An authoring system for computer-assisted instruction in Chinese. In *Proceedings of the 1986 International Conference on Chinese Computing*, 40-46.

Culley, G. (1979). Two pronged error analysis for computer-based instruction in Latin. In *University of Delaware Symposium on language and linguistics*. University of Delaware.

Gale, L. (1983). Montevidisco: an anecdotal history of an interactive videodisc. *CALICO Journal, 1:1*, 42-46.

Kramsch, C., Morgenstern, D., & Murray, J An overview of the MIT Athena language learning project. *CALICO Journal*, June, 1985, 31-34.

Levin, D. (1973) Computer-based analytic grading for German grammar instruction. Stanford University Computer Science Department Technical Report #STAN-CS-73-343, 1973.

Marty, F. (1981). Reflections on the use of computers in foreign language acquisition. *Studies in Language Learning*, 3:1, 26-52.

Morgenstern, D. (1986). The Athena language learning project *Hispania*, 69, 1986, 740-745.